(12) UK Patent Application (19) GB (11) 2 063 532 A

(21) Application No 8029863
(22) Date of filing
16 Sep 1980
(30) Priority data
(31) 85909
(32) 18 Oct 1979
(33) United States of America (US)
(43) Application published
3 Jun 1981
(51) INT CL³ G06F 13/00
(52) Domestic classification
G4A MX
(56) Documents cited
GB 1547381
GB 1496780
GB 1496779
GB 1359662
GB 1353770
GB 1167762
(58) Field of search
G4A
(71) Applicant
Storage Technology Corporation
PO Box 98
Louisville
Colorado 80027
United States of America
(72) Inventor
Barry B White
(74) Agents
Reddie & Grose
16 Theobalds Road
London WC1X 8PL

(54) **Data storage system for a computer**

(57) The system includes one or more host processor interfaces including controllers 12 connected to a main high speed "cache" memory 22 and to disk store interfaces 16 including local controllers, and a supervisory control processor 24.

The system responds to requests by the host computer to store and to retrieve information in the format of a conventional tape drive assembly. The data, however, is actually stored or retrieved in piecemeal form from a plurality of disk storage drives 20_a to 20_n so that the data management system functionally mimics a tape drive and accepts and produces data in serial (tape) format, but in fact stores the data in available spaces in disk units.

Data passes in each direction through a main memory 22 under the control of a processor 24 which allocates blocks of a "tape" file to the disk drives. Interfaces 12_a to 12_n between the computer 10 and memory 22 can contain data buffers and interface between serial format on the computer side and parallel format on the memory side. Interfaces 16_a to 16_n between the disk drives 20 and memory 22 can also contain data buffers and interface between serial format on the disk drive side and parallel format on the memory side.

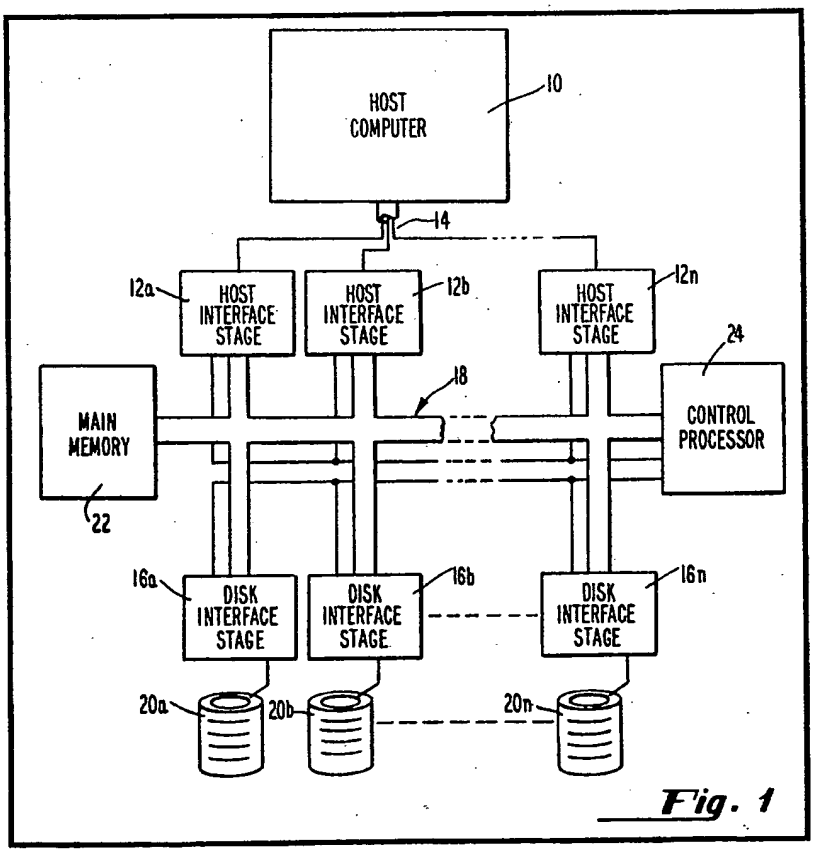

Fig. 1

2063532

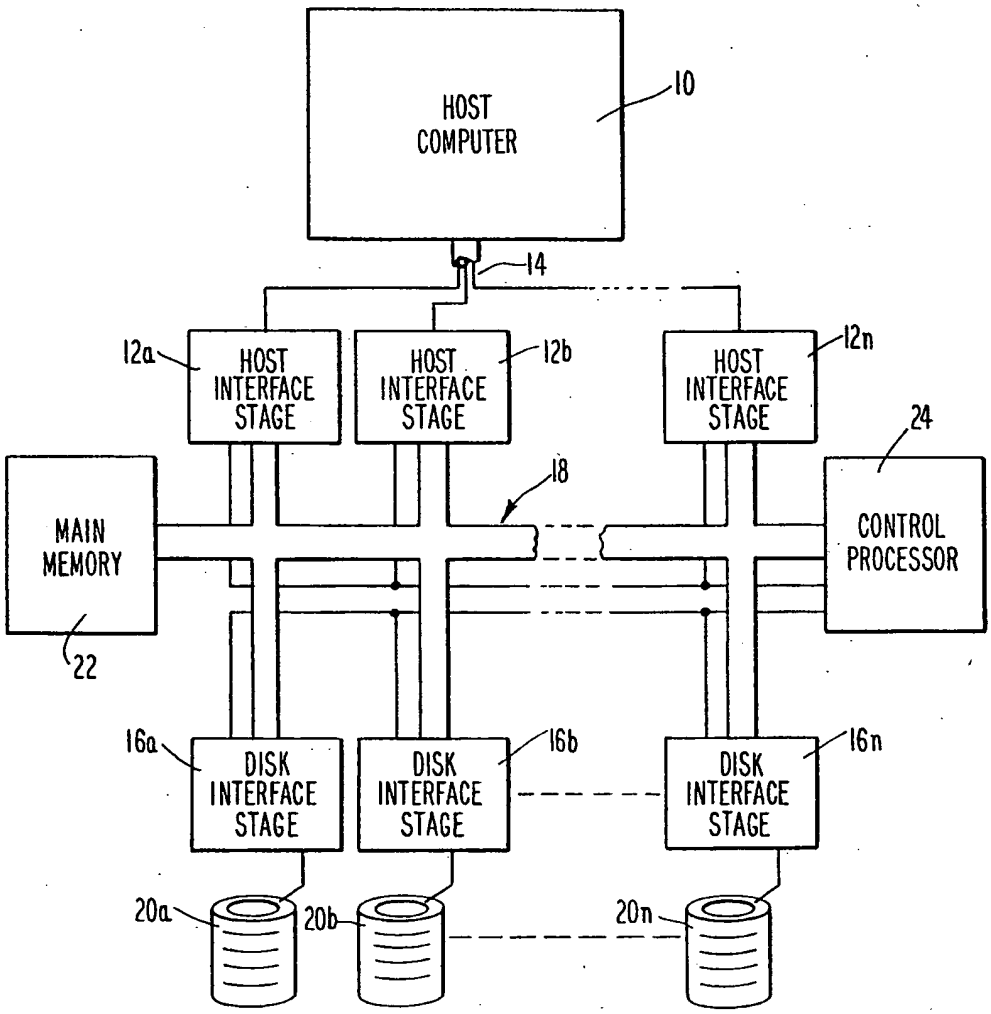

Fig. 1

2063532

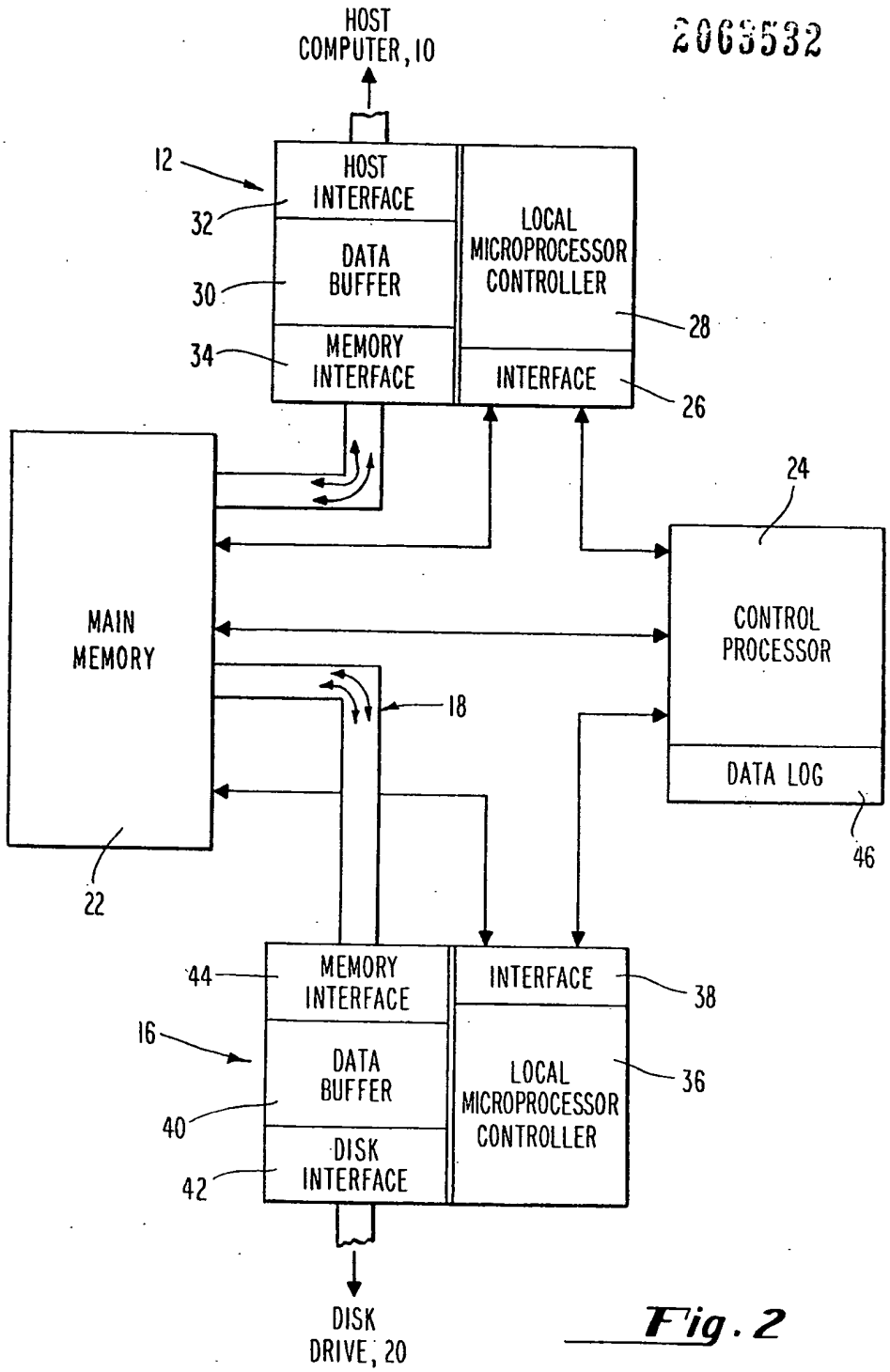

Fig. 2

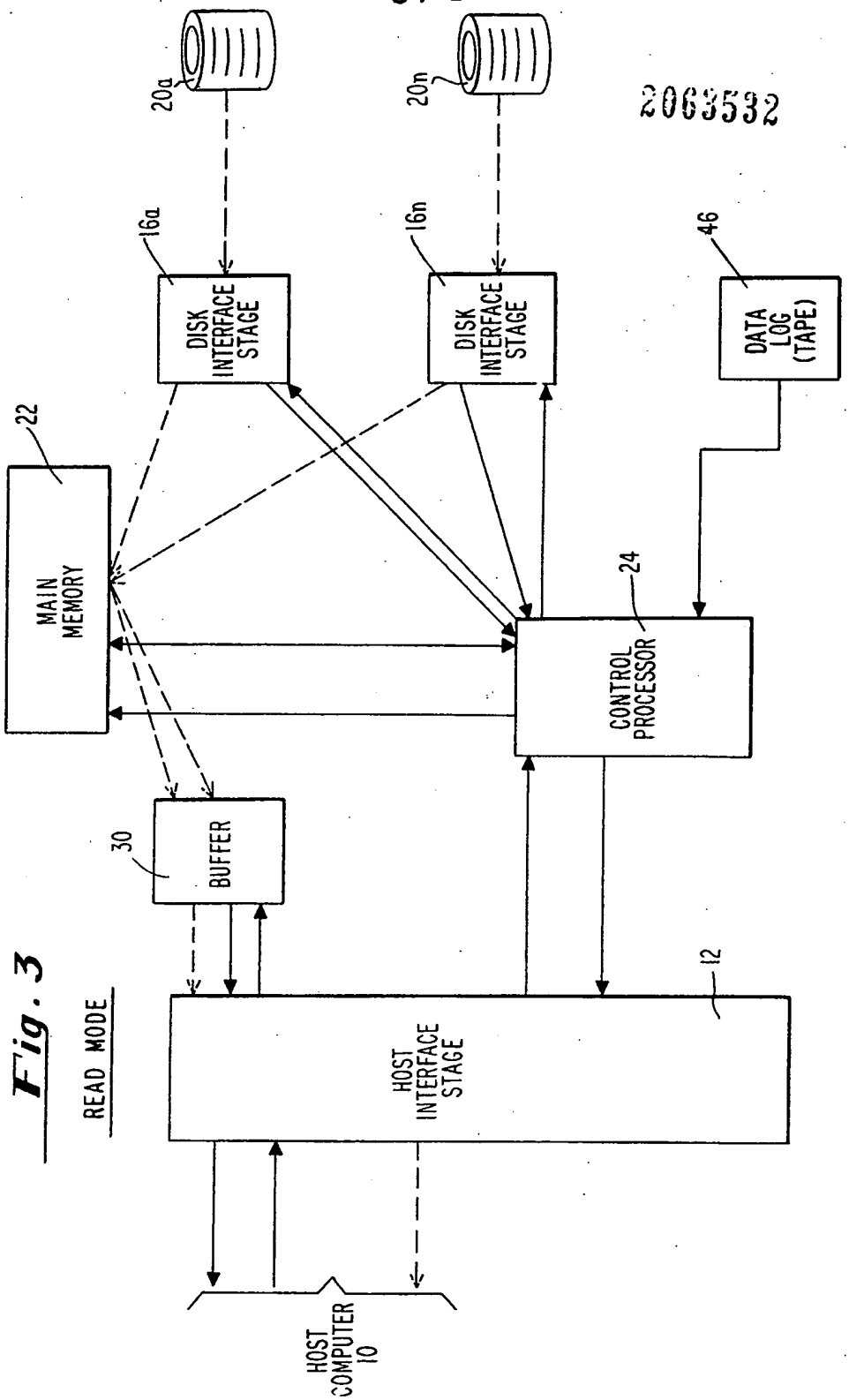

Fig. 3

READ MODE

2063532

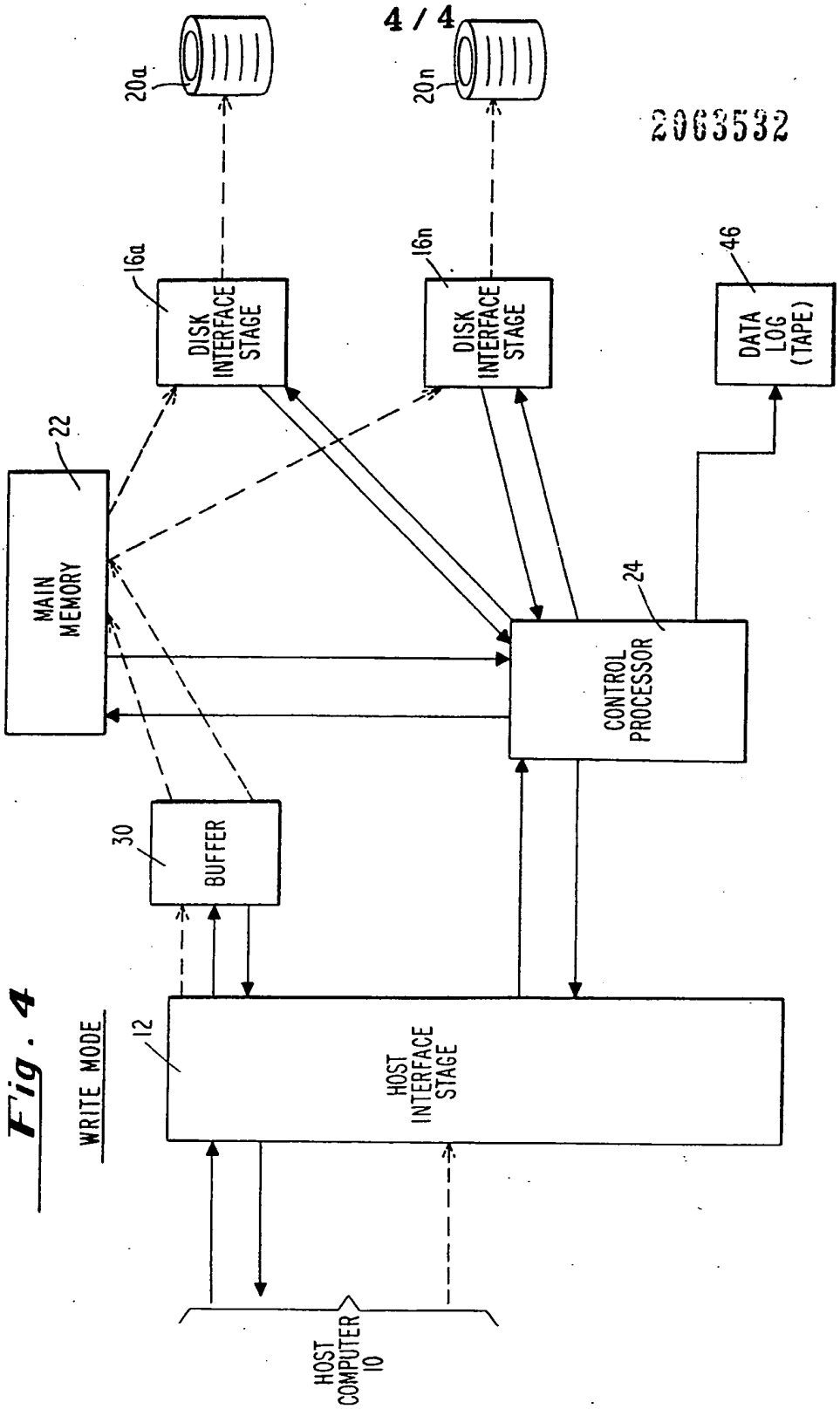

Fig. 4

WRITE MODE

SPECIFICATION

Data storage system for a computer

The present invention relates to a system for storing digital data, and more particularly to an improved system and method for managing and allocating data in a disk storage arrangement.

One of the main considerations in designing and operating a digital data system is the provision of suitable means for storing information received from the central processing unit (CPU) of the computer and providing appropriate information to the CPU when it is requested.

It is known that magnetic tape and disk storage systems have different advantages and disadvantages. Accordingly, each type of storage medium lends itself particularly well to certain types of applications, and not to others. In particular, data files which are serially organized, and which are to be serially accessed, are most compatible with tape drive technology, inasmuch as information is necessarily written upon and read from tape in serial fashion. Disk memory systems, on the other hand, generally have the capability of having information written upon and read from practically any portion of the disks almost instantaneously. As a consequence, multiple data sets all stored on the same disk can be active at once, which is not practical with tape. Therefore, it would always be preferred to use disk memory systems, except for the fact that disk memory is much more expensive than tape.

As a result, while serially-organized data can conveniently be continuously written upon and read from a tape, often writing or obtaining data which is not serially organized presents a problem. If the data is located upon a distant portion of the tape, the tape must be wound on or rewound to the point at which the data is located. A second and far more serious drawback of tape memory is the fact that, for most large-scale applications, the number of reels of tape required to store information is far in excess of, perhaps hundreds of times greater than, the number of tape drives or mounts available to read or write from the tapes. In fact, many large scale data processing operations make use of thousands of reels of tape. In general, some 90% of all tapes contain but one data set (e.g. a customer list or a payroll file) since as a rule the physical characteristics of tape make it impractical to read or write from or to two or more data sets at once, which is very often desirable. Clearly then, tape storage is very inefficient, since only coincidentally will the length of a given data approximate the available memory area. In fact, studies show that normally only about one percent of tape is actually used.

Obviously, no user can provide a tape mount or drive for each reel of tape; rather, individual reels are constantly brought from storage, mounted for use, then demounted and replaced in storage. This is time-taking and inefficient, since it involves the intervention of a human operator.

Efforts have been made to overcome this shortcoming of tape media, for instance by providing a large number of small cartridges of tape and an automatic mechanism to rapidly remove the cartridges from a storage rack and place them on a tape mount. Automatic mounting devices have also been made for use with conventional tapes. To date, however, no completely successful method exists for making large quantities of tape immediately available to tape drives.

At first glance, disk storage system would appear to overcome the shortcomings of tape. Practically any portion of any disk can be assessed very rapidly, so that a disk system does not have to scan through all of the data recorded thereon in serial fashion. Accordingly, by responding to address codes, a disk drive can access a given block of data almost instantly. Thus, multiple data sets can be stored on a single disk and be concurrently processed. However, the cost of disk drives is comparatively great, and moreover available disk drives do not make highly efficient use of the potential storage area.

Studies have demonstrated that at any given time in a disk system, only approximately 75% of potential storage space is actually available for data storage purposes. The "overhead", or total storage space used to support the read/write activity needed to maintain and log data, is thus approximately 20% to 25%. In addition, for most applications, only about two-thirds, or 45% of the available data space is actually used to store data, due to the space requirements of the addressing systems commonly used. Hence, the total utilization factor of most current disk systems is only about 50% or less. Therefore disk systems, although offering attractively high data access and retrieval speeds, are relatively cost inefficient, and may therefore be expensive in terms of the storage space which is actually used. With tape, only one address need be provided per data file, thus greatly increasing the storage efficiency.

It will therefore be seen that it would be highly advantageous to provide a system which exhibits the storage space efficiency of the tape medium and the rapid-access characteristic of disk storage. Desirably, such a memory system would not require any modification to the remainder of the computing system involved; that is, it would "look" to the computer either as tape or disk memories do in the prior art.

It is therefore an object of the present invention to provide an improved storage sys-

tem for a digital computer. The invention is defined in the claims.

In one embodiment of the invention, a "virtual storage system" comprises a host interface stage responsive to commands from a host computer for receiving and transmitting information in a serial, tape form, and a first data buffer within the host interface for temporarily storing data received from the host computer or directed thereto. A plurality of disk storage units are also provided, along with disk storage interfaces and associated second data buffers which temporarily store data directed to and received from the disk storage units. The interface stages are coupled together, and also coupled in common to a large, main memory which receives and queues information flowing from one interface to the other. Finally, a master control processor is coupled to the interface units and to the main memory and directs the operation of the apparatus so that the host interface stages respond to the host computer in the same manner as would a tape system, while the disk interface stages read and write data upon disk memories in random fashion in accordance with currently available memory space.

The invention will be described in more detail, by way of example, with reference to the accompanying drawings in which:

Figure 1 illustrates the overall organization of a virtual storage system embodying the invention;

Figure 2 is a schematic diagram illustrating the flow of control and information signals within the system of Fig. 1; and

Figures 3 and 4 direct the flow of command signals and information in "read" and "write" situations, respectively.

Fig. 1 shows in simplified form the basic architecture of the virtual storage system embodying the invention. The system provides means for controlling one or more extremely fast and accessible disk memory units to accept data and respond in precisely the same fashion as a tape drive, while providing much better use of available storage, thus in effect having vastly expanded storage capability and a far more rapid response time than either tape or disks. The host computer 10 may be of any appropriate type, although it is anticipated that economical operation of the present invention will dictate it use with a large mainframe host computer such as IBM System/360 and System/370 computers, and IBM Model 3031, 3032 and 3033 processor complexes (IBM is a Registered Trade Mark). The virtual storage system can also be used with a plurality of host computers. The storage system comprises a bank of host interface stages 12_a to 12_n, the number of interface stages being dependent upon the amount of data to be transferred. Since the host interface stages are identical, the actual number which is to be used is for present purposes immaterial. The host interface stages are coupled to the host computer 10 by means of conventional channels 14 and receive and transmit information to the computer 10 in the ordinary, serialized format which is associated with tape drives.

One or more disk interface stages 16_a to 16_n are provided, and are coupled to all of the host interface stages by means of a common information bus 18. In this manner, information can be between any given host interface stage and any disk interface stage.

The disk interface stages are each coupled to respective ones of disk memories 20_a to 20_n. As will be discussed in further detail, each disk interface storage comprises a memory interface, a data buffer, and a disk interface, along with a local microprocessor controller for operating the various elements of the interface stage.

Also connected to the main bus 18 is a main memory 22. Memory 22 forms a repository for information flowing from the host interface stages to the disk interface stages, and may further serve as a memory for a control processor 24 also coupled to the main bus 18. The control processor 24, which is the central processing unit or CPU of the virtual storage system, is coupled to each of the host and disk interface stages 12 and 16 respectively, and directs the operation thereof so that information may be received, queued, organized, and stored in the proper manner. The control processor 24 further instructs the various host interface stages 12 to cause them to respond to the host computer 10 in a manner which stimulates a tape drive.

Turning now to Fig. 2, there is shown in further detail the structure of exemplary elements of the system. Processor 24 is coupled directly to host interface stage 12, and in particular to an internal interface 26 which simply facilitates the transfer of signals between the control processor 24 and a microprocessor 28 which forms the controller for the host interface stage. Also comprised within the host interface stage are a data buffer 30, a host interface 32 which makes the data buffer compatible with the host computer 10, and a memory interface 34 through which the data buffer communicates to data bus 18.

The function of the host interface stage 12 is to simulate a tape drive system to the host computer 10, that is to convert the host computer's commands to tape drives into commands to the virtual storage system and to send to the host computer data in the form in which it would be sent by a tape drive. More particularly, the host interface stage 12 accepts the signals from the host computer 10 which are of the type used to operate the tape drives. Such signals are obtained by coupling the host interface stage to the byte multiplex, or block multiplex, or selector chan-

nels of the host computer 10. It will be appreciated that these signals include operator commands such as "mount tape" and "demount tape" as well as machine commands such as "read", "write", and "forward space file". The host interface stage responds to both kinds of signals as if it were an operator and a tape drive, acknowledging the signals, responding that "mounting" of an imaginary (virtual) tape reel has been accomplished, and so on.

The data buffer 30 of each host interface stage accepts data from the host computer 10 in serial form, usually nine bits in parallel, in the precise manner that it would be reapplied to a tape system for writing upon a tape. The buffer 30 in conjunction with the host interface 32 during a write operation de-serializes the information, that is stores it in parallel fields holding the individual bits until eight entire bytes are available. Typically 72 bits are transmitted at a time thus reducing transmission time. In this manner, up to 90% compression of the time required to transmit the data on the line 18 can be achieved. When data is to be exchanged between the buffer 30 and the bus 18, the data flow through the memory interface 34, again in eight parallel bytes so that an extremely rapid exchange of data may take place. The size of the buffer can vary greatly, depending on system requirements but in a preferred embodiment is large enough, 64,000 bytes, to hold an entire record, which expedites time sharing of bus 18.

Ultimately, data received from the host computer 10 is written on to magnetic disks mounted on one or more disk drive units 20_a to 20_n. Each disk interface stage 16 couples the data bus 18 to a disk drive 20 (not shown), and is composed of a local microprocessor controller 36 which operates the various elements of the interface stage in accordance with instructions from the control processor 24. As was the case with the host interface stages 12, with disk interface stages 16 instructions from the control processor are transmitted through an interface 38 to the local controller 36. The latter then responds by causing a data buffer 40, disk interface 42, and memory interface 44 to transfer data to and from the disk drive.

In particular, the memory interface 44 serves to receive eight bytes of data from the bus 18 and serialize the data so that it is installed in the buffer 40 in a single stream of bits. Arrangement and queueing of buffer storage is accomplished by the local controller 36, as is the operation of the disk interface 43 which passes data from buffer 40 to the disk drive at the appropriate time. Still further, the disk interface stages serve to seek and log the locations of data upon the various disks of the disk drive associated therewith, so that the information can be retrieved when needed.

Between the times that data is in a host interface 12 and is received in a disk interface 16, it is stored in the main memory 22. The high speed main memory 22 thus acts as a "bank" or "cache" which holds the data until one of the disk drive units 20 is ready to accept it. When this occurs, the disk interface stage 42 associated with the disk drive 20 signals control processor 24 of its availability. The control processor 24 then instructs the main memory or cache 22 to discharge data to the disk interface stage 42 by way of the bus 18.

In like manner, when the host computer 10 is seeking data, the identity of the data sought is transferred through the host interface stage 12 to the control processor 24, which then determines if the requested data has been previously moved into the main memory. If it has not, the control processor 24 in turn will apply a "transmitter" signal to the disk interface stage 16, and the local controller 26 therein will cause data to be read out of the associated buffer 40, through the associated memory interface 44, and on to the data bus 18, and thence to the main memory 22.

At the same time, the local controller 28 of the available host interface stage 12 will enable the associated memory interface 34, so that the newly-read data will be received from main memory 22 and gated into the host interface data buffer 30. On the other hand, if this has already been done, the data will be immediately gated out of the buffer 30, serialized by the host interface module 32, and transmitted to the host computer 10.

In an alternative embodiment, both the host buffers 30 and disk buffers 40 may be dispensed with, their functions then being performed by main memory 22.

In this fashion, segments of data which had been dispersed throughout a number of disk drives are compiled, queued, and then reassembled automatically into serial form. Accordingly, the information flowing to the host computer 10 from the various disk drives 20 appears in serial form, precisely as if it were being read from a tape. In the foregoing manner, the high speed main memory provides a cache from which data can be selected and queued for subsequent reassembly into serial form before being directed to the host computer.

As will be recognized by those skilled in the art, the various elements of the present virtual storage system may be assembled from commercially-available elements, and coupled together in any convenient fashion. For instance, while all of the various elements of disk interface stage 16 are illustrated as being disposed in one location, remote from control processor 24, in fact the elements may be disposed at different locations, and coupled by means of appropriate cables, buses, or the like. The local microprocessor controllers 24

used for operating the components of each interface stage need only be of rather limited capability and may be any of the various high speed microprocessors available on the market. An example of such microprocessors is the LSI-11 marketed by DEC Incorporated of Boston, Mass, alternatively, a suitable unit can be assembled rom the AMD Co's 2900 series of parts. In like manner, the host and disk interfaces 12 and 16 which accomplish the serializing and deserializing of information may be standard units, such as the model 370 block multiplexer available from the IBM Corporation of Armonk, New York. In like manner, an IBM block multiplexer may be used for direct memory access, the actual connection of the various units being well understood by those skilled in the art.

Similarly, the buffers 30 and 40 used for temporary storage of data in the host and disk interface stages 12 and 16 may be of any appropriate type, although in the embodiment at present preferred, a memory of at least 64K bytes is preferred. One commercially available buffer of this type is manufactured by Fairchild Semiconductor, and comprises a N-MOS random access memory having an access time of 200 nanoseconds.

The high speed memory 22, which serves as a data cache should be of the type generally designated a fast access memory, i.e. one having a cycle time of 400 nanoseconds or less. In a preferred embodiment, the high speed cache has a capacity of 16 megabytes. One commercially-available memory appropriate for use with the present invention is manufactured by Intersil Corp. Sunnyvale, California, and is marketed by ourselves.

Fig. 3 is a functional illustration of the operation of the system in a "read" mode wherein the host computer 10 has requested information from memory. Control signals are shown by solid lines, and data flow by dashed lines. Accordingly, the host computer 10 produces a command signal to the human operator requesting certain information (e.g. "Mount Tape N"), and since it is assumed by the host computer 10 that the data is stored on tape in serial form, only the initial identification or "label" need by specified. Other elements of the virtual storage system, particularly the control processor 24, respond to the information by accessing all data which corresponds to the requested information and causing the data to be assembled in serial form.

Thus, the host computer 10 simply calls or Tape N which, it assumes, according to information supplied by the individual program being run on the host computer 10, contains a particular data set. In fact, according to the virtual storage system of the invention, this particular data set may be spread over a number of different disks and/or may be in a number of locations on a single disk. Therefore, the virtual storage system of the invention responds to an operator command such as "Mount Tape N" by searching through its own memory for data indicating where the various parts of the record which the host computer has denominated "Tape N" have been stored. Thus, the response of the system is not to direct an operator to mount Tape N, but merely to recall the information as to the location(s) of the data set identified as "Tape N" from its memory.

When the host interface stage 12 indicates that a "Mount Tape N" instruction has been completed, which is practically instantaneous in comparison to the mounting of an actual tape, the host computer 10 outputs a command to the interface 12 designating the specific identity of the block of information to be read. This command is transmitted directly by the host interface stage 12 to the control processor 24 which identifies the various areas in which the information is stored. This may be accomplished through the use of a data log 46 which is coupled to the control processor. In the preferred embodiment of the data log 46 comprises a random access memory (RAM) (which may be "backed-up" by a tape drive unit and associated tape in case of loss of memory in the RAM) on which the location of the various sub-blocks of data which together constitute the data requested by the host computer 10 are recorded.

When the locations of the various sub-blocks of data have been identified to the control processor 24, signals are supplied to the controller portions of the various disk interface stages 36, whereupon data is accessed from the various disk storage units 20. The data is then read out into the buffer 40 of the interface stage 16 and prepared for transmission to the main memory cache 22 for temporary residence there until needed by the various host interface stages 12.

When the buffer 40 is filled with data, a signal is output from the disk interface stage 16 to the control processor 24 indicating this fact, at which time the control processor 24 transmits the buffered information directly to a previously-allocated space in the main, high speed memory cache 22.

With prior art tape systems, the initial "Mount" operation ordinarily occupies approximately 30 sec.-5 min. The total time required to execute a "read" command varies between 1-10 milliseconds depending on the characteristics of the individual device and of course on the length of the record being read. In a preferred embodiment of the illustrated virtual storage system, a "Mount" operation tyically occupies less than one second. The actual "read" operation typically occupies 1-10 milliseconds as before, depending on record length and the characteristics of the cache 22. At this point, therefore, the information to have been read will have been very rapidly placed in the cache 22 where it is

stored until a signal from memory interface 12 indicates to the cache that the host computer is ready to receive the data. At this stage, then, the data has been gathered from the various points in the various disks 20 where it has been stored, has been put into serial form in the proper order, and is ready to be read into the host computer 10 with no delay for address time or for searching for the information on the various disks. Clearly, therefore, the memory system offers significant time advantages over disks; and as compared to tape systems, avoids the mounting time delays completely. In this way, the "anticipatory buffering" provided by the use of the cache combines the advantages of disk and tape in that it combines the lack of mounting required for disks with the serial format of tapes, thus saving time to the host computer by having the data completely organized in the cache. At the same time, owing to the superior organization and cooperation of the various elements, very little of the actual disk storage is dedicated to directing read/write activity or maintaining or logging the data thereon, sometimes referred to as the "overhead" factor in disk memory capacity. Accordingly, a much higher percentage of actual disk memory is available for data. In this way, the storage efficiency of disk memory can be upgraded to a level similar to that of tape.

In the host "write" mode illustrated in Fig 4, the procedure is substantially the reverse of the "read" mode just described. As in Fig. 3, control signals are shown by solid lines and the flow of data by dashed lines. When the host computer 10 signals, through conventional tape drive codes, that it desired to "write" or record data, this command is transmitted through interface stage 12 to the control processor 24. The latter instructs interface 12 to acknowledge the command, and at the same time interrogates the disk interface stage 16 in order to find a buffer 40 which will accept some or all of the data. When such locations are determined, they are logged in data log 46 by the control processor 24.

The control processor 24 then directs the host interface stage 12 to accept the data from the host computer 10 and direct it to appropriate locations in the data buffer 30 of the host interface stage 12. Should the buffer 30 become filled before corresponding space is available in the buffers 40 of the disk interface stages, an "allocate space" instruction is transmitted by the control processor to the memory cache 22, after which data from the host interface stage buffer is applied to the memory cache 22. At some subsequent time, when adequate buffer capacity is available in one or more disk interface stages 16, the control processor instructs the cache memory 22 to supply the stored data to the appropriate disk interface stage buffers 16 for disposition upon the disks of the associated disk drives 20.

As each block of data is applied to a host interface 32 by the host 10, the memory interface module 34 determines where in the associated buffer 30 the newly-received information will be stored; compresses the data in serial fashion so as to produce, for instance, eight channels of data; and adds an identification bit or bits to the information block indicating the size of the block. In this manner, data is continually fed to the buffers 30 until a given block of data is filled or until the host computer generates a signal indicating the end of a given record. It is an important subsidiary feature of the present invention that such a signal is interpreted by the local controller 28 as an end of message signal, subsequent to which no more buffer memory space need be allocated.

At this point, the control processor 24 directs the data now stored in data buffer 30 to the high-speed cache 22. During the time the host computer is storing data in data buffer 30 as described above, the control processor 24 can be "looking for" sufficient space upon disk drives 20, and can allocate that space to the data then being stored in data buffer 30. Thus, when the end of record signal is received in the control processor, data it can be serialized and fed into the high speed cache, as a temporary storage space, and is then passed to disk drives 20 for storage.

Put differently, when the host computer produces an initial "Mount Tape" command (i.e. directs the operator to provide a blank tape for data storage purposes) this command is passed through the host interface stage 12 to control processor 24. The control processor 24 responds to the "Mount Tape" instruction by seeking space upon one or more disks 20, and by reserving adequate space in one or more of the disk interface stage buffers 30. This done, the control processor 24 instructs the host interface stage 12 to accept data from the host 10 and to deseriablize it and store it in the associated buffer 30. Once this buffer is full, if no more buffer space is available, this data is passed to the memory cache 22. When sufficient disk space is available, the next step is for the control processor 24 to instruct the disk control interface 42 of the disk interface stage 16 to write the data upon the associated disk device 20 after which the interface module 44 reads and serializes the information, placing it in its associated buffer 40, whence it is written onto the associated disk 20. It will be understood that another possibility is to have the data copied or stored onto a conventional tape drive, if for example duplication be deemed desirable; that is, the data storage system of the invention can be paralleled with a conventional prior art tape system, if desired.

It will be apparent to those skilled in the art that the association of a high speed cache

with a control processor, along with host interface and disk interface stages and associated control modules permits the combinations of the advantages of prior art tape and disk drive systems; that is, the ready accessibility of disk drives is combined with the storage efficiency of tape drives to yield a fast high efficiency "virtual storage system" in which a disk system is made, through the intermediate of the invention, to look to the central processor like a tape drive, but without the drawbacks thereof. Since the memory disks are permanently mounted, there is no need for an operator's intervention when a tape-type record (i.e. a sequential record) is sought. Moreover, prior art art disk drives can be used with the virtual storage system of the invention without modification; that is, the system of the invention can be viewed as a unit or interface inserted between a prior art host computer and prior art disk drives. Therefore, there is no need to modify the disk drives or the host computer in order that they may be operated in conjunction with the virtual storage system of the invention.

Furthermore, it will be appreciated that the virtual storage system of the invention can be used to interface between any number of host computers and any number of disk drives. Therefore, it will be appreciated that should an operator have more than one host computer and a limited number of disk drives, the virtual storage system of the invention can be used to achieve maximum storage efficiency and eliminate undue multiplication of disk or tape storage systems. The host interface units of the invention may be coupled to more than one host computer. Moreover, it will be appreciated that the provision of a high speed memory or cache is essential to the "anticipatory buffering" which is a very important feature. By using the high speed cache as an intermediate buffer between the host computer interface and the various disk drive interface stages, buffering of multiplex data can be accomplished; that is, data stored in a number of places upon a number of disk drives (referred to as "in random form") can be assembled, sequenced, and stored temporarily in the high speed cache until it is called for by the host computer. In this way, there need be no delays in reading the information from memory into the host computer. In a similar fashion, data output from the host computer can be arranged, divided, and stored in the high speed cache until such time as various disk storage areas are available to store the information.

A particularly significant possibility which is provided by the storage system is that of data compression. Heretofore a common scheme of data compression, which may involve the replacement of a long string of digital "1's" or "0's" with symbols indicating the length of the string, had not been useful with disk drive units, since this scheme compreses addressing information as well as data. However, since according to the present application disk drives appear as tape, this difficulty of the prior art is obviated, and data compression is made compatible with disk storage. Preferably, data compression is implemented in the host interface stage; that is during a write operation long continuous strings of "1's" or "0's" may be detected and replaced with shorter symbols; during a read operation, these are detected and replaced by the thus-defined data.

Finally, it will be appreciated that there are a number of other modifications and refinements which can be made to the virtual storage system of the invention and that the examples discussed above are exemplary only.

CLAIMS

1. A data storage system for use with a host digital computer, comprising at least one host interface responsive to commands from the host computer for storing or accessing information arranged in serial form, the or each host interface comprising a first data buffer for storing data received from or directed to the host digital computer; a main memory; at least one disk storage unit comprising a plurality of disks adapted to store digital information thereon, the or each disk storage unit having an associated disk interface with a second data buffer coupled to the disk storage unit and to the main memory for storing data received from or directed to the disk storage unit; and a control unit coupled to the interface for causing the or each host interface to receive and accept data in serial form, and for causing the or each disk storage unit interface to store and to access information in random form.

2. A data storage system for use in conjunction with one or more host computers and disk drives, comprising a data cache, comprising a high-speed main memory unit, a number of host computer interfaces at least equal to the number of host computers, each comprising a local controller and interface means to couple the host computer interfaces to the host computer(s) and to the data cache, a number of disk drive interfaces, each comprising a local controller and interface means to couple the disk drive interfaces to associated disk drives and to the data cache, and a controller for coordinating the operation of the host computer interface stages, the disk drive interface stages, and the data cache.

3. A data storage system as claimed in either of claims 1 and 2, wherein the system is responsive to both operator and machine commands.

4. A data storage system as claimed in either of claims 1 and 2, further comprising means for effecting data compression and data decompression.

5. A data storage system as claimed in claim 2, wherein each host computer interface comprises a data buffer.

6. A data storage system as claimed in claim 2 or 5, wherein each disk drive interface comprises a data buffer.

7. A method of storing digital data, comprising receiving the data in a main memory; determining appropriate storage locations on disk memory units; transmitting the data to disk interface units associated with said disk memory units; and writing the data from the disk interface units on to the disk memory units.

8. A method according to claim 7, wherein the flow of data between the disk memory units, and said main memory is controlled by a system controller.

9. A method according to claim 8, wherein the controller responds to operator as well as machine commands.

10. A method according to claim 7, wherein the data is received from a host computer by a host interface unit and is de-serialized prior to transmission to the main memory, and is re-serialized prior to being written on to the disk memory units.

11. A method according to any of claims 7 to 10, further comprising the step of data compression.

12. A method according to any of claims 7 to 11, wherein the addresses of the storage locations at which the data is stored on the disk units are stored in a random access memory.

13. A method according to claim 12, wherein the addresses are additionally stored in back-up memory means.

14. A method for retrieving data stored on a plurality of disk memory units, comprising determining where the data is stored; causing the data to be retrieved from the disk memory units, and written into a main memory; causing the data to be queued in the main memory; and causing the data to be transmitted to a host computer.

15. A method according to claim 14, wherein the data is de-serialized prior to being written into the main memory, and is re-serialized prior to being transmitted to the host computer.

16. A method according to claim 14 or 15, further comprising the step of data decompression.

17. A method according to claim 14, 15 or 16, comprising the step of storing the data in a buffer interposed between the main memory and the host computer.

18. A method according to any of claims 14 to 17, wherein the flow of data between the disk units, the main memory and the host computer is controlled by a system controller.

19. A method according to claim 18, wherein the controller responds to operator as well as machine commands.

20. A method for storing digital data, comprising the steps of receiving serially organized data from a host computer in host inter ace means; determining what portions of disk memory means are available for storage of said data; dividing the data into blocks corresponding to the said portions; and storing the data in block form in the said portions.

21. A method according to claim 20, wherein the data is de-serialized in the host interface means.

22. A method of claim 20 or 21, wherein the data is stored in a main memory prior to being stored in the said portions.

23. A method according to claim 21, wherein the data is compressed in the host interface means.

24. A method according to any of claims 20 to 23, wherein the addresses of the locations at which the data is stored are stored in random-access memory.

25. A method according to any of claims 20 to 24, wherein the serially-organized data is additionally stored on magnetic tape memory means.

CLAIMS (7 Jan 1981)

1. A data storage system for use, in conjunction with one or more host computers and disk drives, comprising;

a data cache, the cache including a high-speed main memory unit,

a number of host computer interfaces at least equal to the number of host computers, each comprising a local controller and interface means to couple the host computer interfaces to the host computer(s) and to the data cache,

a number of disk drive interfaces, each comprising a local controller and interface means to couple the disk drive interfaces to associated disk drives and to the data cache, and

a controllers for coordinating the operation of the host computer interface stages, the disk drive interface stages, and the data cache.

2. A data storage system for use with a host digital computer, comprising:

at least one host interface responsive to commands from the host computer for storing or accessing information arranged in serial form, the or each host interface comprising a first data buffer for storing data received from or directed to the host digital computer;

a main memory coupled to the or each host interface for storage of data;

at least one disk storage unit comprising a plurality of disks adapted to store digital information thereon, the or each disk storage unit having an associated disk interface with a second data buffer coupled to the disk storage unit and to the main memory for storing data received from or directed to the disk storage unit; and

a control unit coupled to the interfaces for

causing the or each host interface to receive and accept data in serial form, and for causing the or each disk storage unit interface to store and to access information in random form.

3. A data storage system as claimed in either of claims 1 and 2, wherein the system is responsive to both operator and machine commands.

4. A data storage system as claimed in either of claims 1 and 2, further comprising means for effecting data compression and data decompression.

5. A data storage system as claimed in claim 2, wherein each host computer interface comprises a data buffer.

6. A data storage system as claimed in claim 2 or 5, wherein each disk drive interface comprises a data buffer.

7. A data storage system according to either of claims 1 and 2, wherein the data is received from a host computer by a host interface unit and is de-serialized prior to transmission to the main memory, and is re-serialized prior to being written on to the disk memory units.

8. A data storage system according to claim 7, wherein the serially organized data is additionally stored on magnetic tape memory means.

9. A data storage system according to either of claims 1 and 2, wherein the addresses of the storage locations at which the data is stored on the disk units are stored in a random access memory.

10. A data storage system according to claim 8, wherein the addresses are additionally stored in back-up memory means.

Printed for Her Majesty's Stationery Office
by Burgess & Son (Abingdon) Ltd.—1981.
Published at The Patent Office, 25 Southampton Buildings,
London, WC2A 1AY, from which copies may be obtained.